福井大学
工学部 研究報告
第44巻 第1号
1996年3月

ブール関数の同定問題に対する遺伝的アルゴリズムの適用と遺伝的操作

山口 哲司* 西野 順二** 小高 知宏** 小倉 久和**

An Application of the Genetic Algorithm to the Determination Problem of a Logic Function

Tetsushi YAMAGUCHI, Jyunji NISHINO, Tomohiro ODAKA, and Hisakazu OGURA

(Received Feb. 29, 1996)

In this paper, we focus on a determination problem of a logic function using the Genetic Algorithm(GA), and examine a knowledge expression in GA. If the logic function is expressed by a truth table, we can derive an expression of the formula from the truth table using a logical method, like the Karnaugh map method or the Quine-McClusky method. We apply the GA to this problem, and analyze characteristics of a GA model for knowledge acquisition.

There are several ways to express the formula of a logic function. We use two ways to gene coding methods of logic function, the sum-of-product form coding and the general form coding. We simulate GA using these two gene codings, and consider the characteristics of the gene structure and their effects to the GA from the results. In sum-of-product form coding, we find out that the ratio of true values in truth table make an effect on the difficulty of a problem. In the general form coding, we find out that characteristics of the problem relate to the efficacy of the Building-Blocks in the GA search.

1 はじめに

本研究では，遺伝的アルゴリズム (GA) による知識獲得の応用としてブール関数の同定問題を取り上げ，GA における知識表現の課題を検討する．ブール関数とは，基本的なブール演算である和，積，補元の組合せで表わされる関数である．関数値として 0/1 の 2 値を取り，真理値表によって表現することができる．あるブール関数を真理値表という形で与えられれば，カルノー図法やクワイン・マクラスキー法などの論理的手法により式表現を導くことができるが，本研究では GA をブール関数の式表現を求める問題に適用することにより，GA の知識獲得における特徴について解析する．GA はあくまで最適化手法であるため，論理的手法のように常に真理値表に対して全正解を与える式表現を求めることは難しい．しかし GA は設計しだいでは問題のサイズによる影響の少ないシステムを構築することが可能である．この特徴を活かし，より多変数な問題に適用した場合でも，ある程度の正解を与える式表現を求められるような柔軟性を持ったシステムの構築を目的とする．

* 工学研究科情報工学専攻 ** 工学部情報工学科

ブール関数には，いくつかの式表現方法が存在するが，ここでは積和形コーディングと一般形コーディングの2通りのコーディング方法を定義する．遺伝子コーディングは問題の知識を直接表現する部分であり，遺伝子を評価する適応度，遺伝子を変化させる遺伝的操作と併せてGAにおいて最も重要な部分である．進化の上で的確な形質の継承が行われるためには，遺伝子の構造とその表現する意味を考慮した操作が必要である．また，操作が効率的に機能するためには適応度の設定も重要なファクターの1つである．本研究では，上の2つのコーディング方法による試行を行い，その結果から遺伝子構造の特徴，それらがGAに与える影響を中心に考察し，今後の課題，展望などを検討する．

2 ブール関数の遺伝子表現と遺伝的操作

2.1 積和形によるブール関数の知識表現

ブール関数における式表現の1つに積和形という表現方法がある．積和形は積項の和で構成される式であり，例えば，次に示すような式である．

$$f(x_1, x_2, x_3) = x_1 + \overline{x_1}x_3 + x_1\overline{x_2}x_3 + \overline{x_1}\overline{x_2}$$

積和形は形式的な式表現であるので，遺伝子コーディングが容易で扱い易いという利点がある．一般にGAでは，優良解の形成に必要な部分構造 (ビルディングブロック) の存在が重要である．ビルディングブロックが，世代の経過に伴って子孫に継承，蓄積されることにより徐々に優良解が形成されるからである．積和形では式の構成に必要な積項が，ビルディングブロックとして有効に作用することが期待できる．

(1) ビット列による遺伝子コーディング

3変数のブール関数を例にして，コーディング方法を説明する (図1).

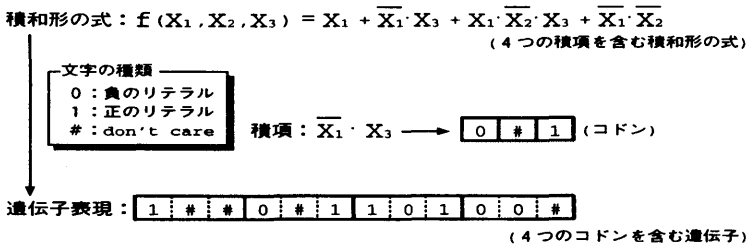

図 1: 積和形の遺伝子表現

積和形コーディングにおいては，0，1，#の3種類の文字を用いて遺伝子を表現する．3種類の文字は，それぞれ0は負のリテラル，1は正のリテラル，#はdon't careを表わす．$\overline{x_1}x_3$という積項の場合は，変数x_1が負のリテラル，変数x_2がdon't care，変数x_3が正のリテラルであるから，0#1という文字列で表わされる。このように，1つの積項を表わす文字列をコドン (codon) と呼ぶことにする．式全体を表わす遺伝子は，コドンを一列に並べた形で表現する．図1の例では，積和形の式は4つの積項で構成されており，遺伝子は4つのコドンにより構成される．問題の変数の数をm，式に含まれる積項の数をnとすると遺伝子長はm × nとなる．したがって，コーディングする式によって遺伝子長は変化する．

(2) コドンを単位とする遺伝的操作

積和形コーディングでは，遺伝子がコドンの集合であるため，遺伝的操作によってコドンの組合せ探索と多種多様なコドンの生成という2つの効果が必要となる．そこで，コドンを単位とする遺伝的操作を定義する．

コドンを単位とする交叉は，具体的には図2のように行う．まず，2つの親遺伝子 (parent1 と parent2) を選択し，それぞれの遺伝子中の全てのコドンに対して，どちらの子遺伝子 (child1 と child2) に継承されるかをランダムに決める．各コドンの継承先に従ってコドンの組み替えを行い，新しい2つの子遺伝子を生成する．したがって，2つの親遺伝子のコドン数の和は2つの子遺伝子のコドン数の和と等しい．

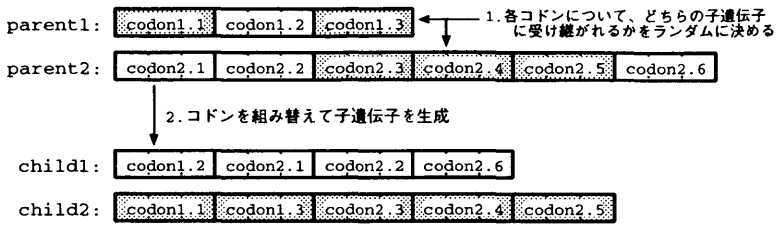

図 2: コドンを単位とする交叉

コドンを単位とする突然変異は，具体的には図3のように行う．まず変異を起こすコドンを遺伝子中からランダムに1箇所選択する (codon3)．そして，このコドンに変わる新しいコドン (codon3′) をランダムに生成し，選択されたコドン (codon3) と置き替える．

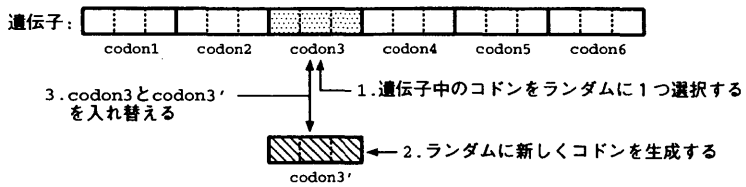

図 3: コドンを単位とする突然変異

2.2 一般形によるブール関数の知識表現

ここでは，例えば次式のように2項演算の AND，OR と単項演算の NOT の組合せにより表わされる式表現を一般形と呼ぶことにする．

$$f(x_1, x_2, x_3) = ((x_1 + x_3) \cdot x_1) \cdot ((\overline{x_2 \cdot x_3}) + (x_1 \cdot x_2))$$

(1) 木構造を用いた遺伝子コーディング

一般形の式において AND，OR，NOT を非終端記号，リテラルを終端記号として図4のように木構造で表現したものを遺伝子とする．

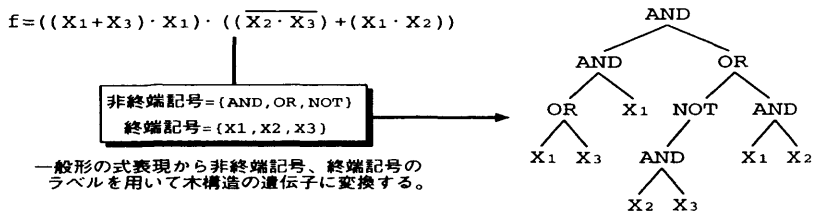

図 4: 木構造による遺伝子表現

(2) 木構造表現に対する遺伝的操作

木構造遺伝子に対する遺伝的操作として，次に示す交叉 (図 5)，突然変異 (図 6) を定義する．これらは，文字列を対象とした場合の操作の自然な拡張である．

交叉では親遺伝子間で部分木の交換を行う．各遺伝子において，ランダムにノードを 1 箇所選択し交叉点とする．そして，交叉点以下の部分木を親遺伝子間で交換する．

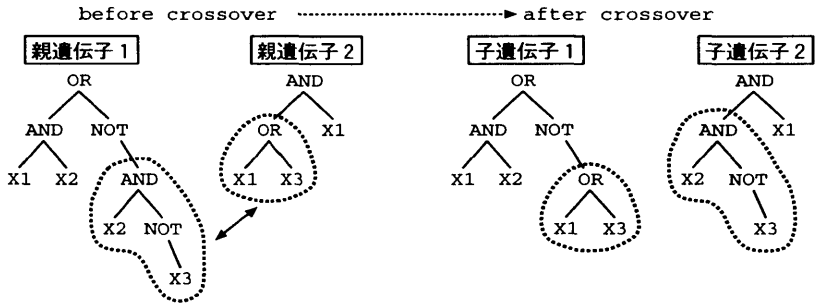

図 5: 木構造で表現された遺伝子の交叉

突然変異ではノードのラベルの変換を行う．遺伝子中のノードをランダムに 1 箇所選択し突然変異点とする．選択したノードのラベルが AND ならば OR に，OR ならば AND に変換する．選択したラベルがリテラルならば，その他のリテラルに変換する．NOT に対する突然変異として，あるノードへの NOT の挿入，あるいは NOT の削除が考えられるが，この方法では変異点以下の部分木の意味が反転し，遺伝子全体の意味が大きく変化すると考えられる．このような理由から，ここでは NOT に対する突然変異は行わないことにする．

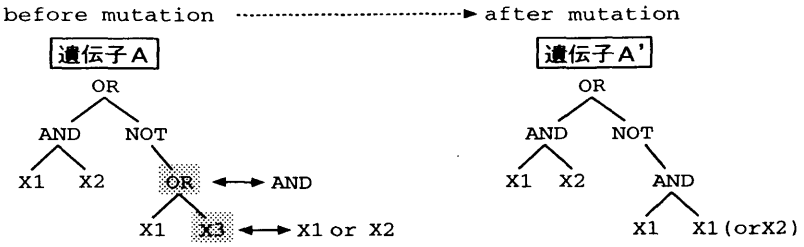

図 6: 木構造で表現された遺伝子の突然変異

3 GA による式表現の獲得シミュレーション

3.1 GA の構成と設定

これまでに説明してきたコーディング方法，遺伝的操作を用いた GA の構成，および処理の手順を図7に示す．

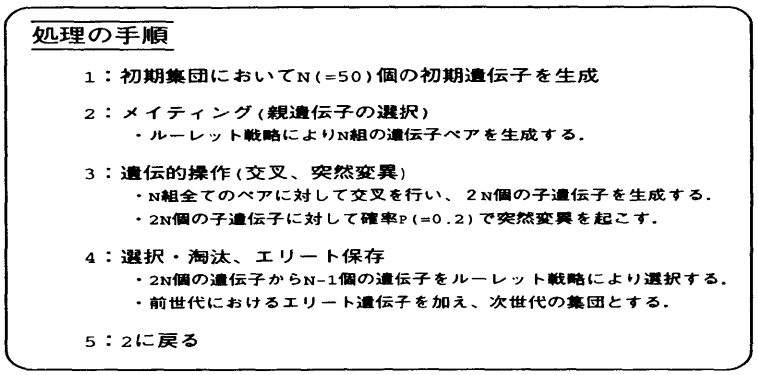

図 7: GA の処理手順

この GA では，問題として与えられた真理値表に対して，より多くの正解を与える式を求めることが目的である．そこで，積和形コーディングにおいては，問題として与えられた真理値表に対する正解数を遺伝子の適応度とする．一般形コーディングにおいては，正解数をそのまま適応度とした場合，世代の経過に伴って遺伝子のノード数が増大する傾向があるので，次にように遺伝子にノード数を考慮した適応度を用いる．

$$\text{適応度} = \frac{(\text{真理値表に対する正解数})^2}{\text{遺伝子のノード数}}$$

遺伝子のノード数を分母にすることによってノード数の多い遺伝子は適応度が低くなり，ノード数の増大を抑制することができる．また，正解数とノード数では正解数を優先する必要があるので，分子を正解数の 2 乗とし正解数の重みを大きくする．

3.2 いくつかの問題に対する試行結果

各コーディング方法による試行を，変数の数，問題の種類をいろいろ変えて行った．本研究では，変数の数は 5，7，9 の 3 種類とし，それぞれの場合において次に示す A～D の 4 種類の問題を用意する．それぞれの変数と問題の組合せに対して乱数のシードを変え，1000 世代までの試行を 10 回ずつ行った．

【問題】

A：多数決関数 (変数値 1 が変数値 0 より多い場合に関数値が 1 となる関数)

B：真理値表における関数値 1 の割合が全体の 25%(ランダムに作成)

C：真理値表における関数値 1 の割合が全体の 75%(ランダムに作成)

D：真理値表における関数値 1 の割合が全体の 50%(ランダムに作成)

各コーディング方法における試行結果をそれぞれ表 1，表 2に示す．表中の値は 10 回の試行における最終的なエリート遺伝子の正解数の平均値であり，括弧の中の値は正解率を示す．

表 1: 積和形コーディングによる試行結果

変数の数	5 変数		7 変数		9 変数	
問題 A	30.2	(94.4%)	110.8	(86.6%)	413.3	(80.7%)
問題 B	31.0	(96.9%)	105.4	(82.3%)	392.5	(76.7%)
問題 C	31.7	(99.1%)	100.4	(78.4%)	389.1	(76.0%)
問題 D	30.9	(96.6%)	97.9	(76.5%)	310.8	(60.7%)
全体の平均	31.0	(96.7%)	103.6	(81.0%)	376.4	(73.5%)

表 2: 一般形コーディングによる試行結果

変数の数	5 変数		7 変数		9 変数	
問題 A	27.7	(86.6%)	126.6	(98.9%)	489.1	(95.5%)
問題 B	27.1	(84.7%)	97.2	(75.9%)	386.8	(75.5%)
問題 C	27.9	(87.2%)	98.2	(76.7%)	388.1	(75.8%)
問題 D	24.2	(75.6%)	80.1	(62.6%)	303.2	(59.2%)
全体の平均	26.7	(83.5%)	100.5	(78.5%)	391.8	(76.5%)

問題の種類別によって正解率に大きな差があり，どちらのコーディング方法においても，ほぼ A，B，C，D の順に正解率が高くなっている．積和形コーディングと一般形コーディングを比較すると，積和形コーディングの方が全体に渡って良い結果を得ることができた．しかし問題 A においては，一般形コーディングの方が極端に高い値を示している．このように問題によって正解率に差が現われるのは，問題における関数値の割合，規則性といった特徴が GA の探索に影響するためであると考えられる．これらのことを検討することにより，各コーディング方法における遺伝子構造の特徴を見出し，知識の表現方法としての問題点を考察する．

4 試行結果に対する考察

4.1 積和形コーディングを用いた場合

次に示す2つの値から，各試行における遺伝子の進化過程の様子を調べる．1つは，遺伝子に含まれるコドンの数である．問題B，Cでは，試行によってこの値にばらつきがあり，問題の特徴が影響していると考えられる．もう1つは，遺伝子の関数値が不正解である場合の数である．これには，問題の関数値0に対して遺伝子が関数値1を与える場合と，その逆の2通りの場合がある．前者の場合をmiss，後者の場合をnoneと呼ぶことにする．これら2つの値が，世代の経過に伴ってどのように推移するかを調べる．

以下では，ブール関数を表記する際に積項をコドン形式で表わした式表現を用いる．

(1) 問題A(多数決)に対する試行結果の考察

この問題に対する試行結果のグラフを図8(7変数)に600世代まで示す．グラフから分るように，進化の過程においてエリート遺伝子のnoneは徐々に減少する．これは関数値1に対する正解数の増加を示している．これに対してmissはほぼ一定の値をとり続け，関数値0に対する不正解数がほとんど変化していないことを示している．この結果，初期の世代ではnoneの方がmissよりも多くを占めているのが最終的には逆転している．

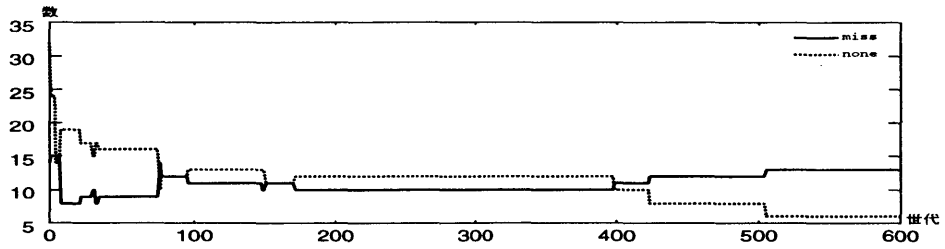

図 8: エリート遺伝子における miss と none の推移 (7 変数)

この試行における最終的なエリート解は，

$$f = 11\#\#1\#\# + \#01\#11\# + \#\#1\#\#11 + 1\#\#1\#\#\# + \#1\#\#011 + 0111\#\#0 +$$
$$1\#10\#\#1 + 0111\#\#1 + \#1\#\#101 + 0\#\#1111 \quad (7\text{変数})$$

という式である．コドン内部にdon't careと1が多く含まれているのが特徴的である．多数決関数の式を構成する際に，このようなコドンが遺伝子に与える影響を考えてみる．例えば，7変数の1番目のコドン11##1##は，16通りの変数値の組に対応し，そのうち1100100以外の15通りに対して正解を得る．また，4番目のコドン1##1###は，32通りに対応し，26通りの場合に対して正解を得る．これらのコドンが遺伝子に含まれることにより，41通りの変数値の組に対する正解が確定的となる．このように，遺伝子の適応度を上昇させるという意味では，これらのコドンは有益である．しかし一方では，7通りの変数値の組に対する不正解も確定する．積和形は，関数値が1になる変数値の組を積項によって表現する構造である．このため，この種のコドンが遺伝子に含まれる限り，不正解の部分が訂正されることはない．つまり，(i) 遺伝子の適応度を上昇させる効果を持つと同時に，(ii) 脱却が困難な局所解に陥る原因でもあると言える．探索の初期段階では，この種のコドンは (i) の

意味で有効に作用するので，世代の経過に伴って集団内に広まり，エリート遺伝子と集団全体の適応度は徐々に上昇する．しかし，これと同時にmissによる不正解も確定的になり，表面化しない部分で(ii)による影響も徐々に進行する．この結果，図8のようにmissとnoneの値に片寄がみられ，最終解は上に示したような式になると考えられる．

(2) 問題B(25%)に対する試行結果の考察

この問題では，コドンを1つしか含まない解を得た試行が多数ある．例えば次のような式である．

$f = 0\#\#1\#0001$ (9変数)

この試行における，200世代までのコドン数の推移のグラフを図9に示す．エリート遺伝子のコドン数は初期世代で徐々に減少し，以後，最終世代まで小さい値で推移している．これに従うように集団の平均も減少し，80世代目に集団の平均が1に収束している．

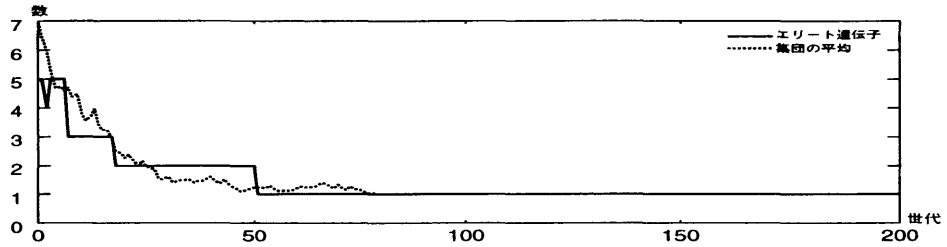

図 9: 9変数の試行における遺伝子中のコドン数の推移

積和形コーディングでは，関数値1の部分において正解を与えるコドンがいくつか集まることで，より適応度の高い遺伝子を構成する．しかし，問題Bのように真理値表において関数値1の占める割合が少ない問題では，遺伝子のコドン数が少ないことにより，関数値0の部分において正解を得ることができる．このため初期の世代ではコドン数が少なく，missによる不正解の少ない遺伝子がエリート遺伝子となりやすい．その他の有益なコドンが生成されないまま，このようなエリート遺伝子が集団で支配的になることにより，図9のような現象が起ると考えられる．

(3) 問題C(75%)に対する試行結果の考察

問題Cは，関数値の割合で問題Bと対称的な関係にあるが，最終解の正解数の平均値はほぼ同じであった．この問題に対する試行で得た最終的なエリート解を次に示す．

$$f = \#\#\#\#1\#\#\#0 + 10\#101010 + \#\#\#\#11\#\#1 + \#\#1\#\#0\#\#\# + \#\#0\#\#\#\#\#\# +$$
$$\#1\#\#\#\#\#\#\# + \#010\#\#\#0\# + 0\#1\#\#1\#\#\# \quad (9\text{変数})$$

遺伝子に含まれるコドンは大部分がdon't careで構成されており，どちらもほとんどの場合で関数値1を与える式である．この試行におけるmissとnoneの推移を図10に示す．進化の過程でnoneは初期の世代で急激に減少し，最終的には0に近づく．これに対してmissは初期の世代で徐々に増加し，その後は横這い状態である．

この問題では関数値1の割合が多く，don't careを多数含むコドンによって構成される遺伝子は関数値1の部分においてほとんど正解を得る．極端な例では，全ての場合で関数値1を与える遺伝子で

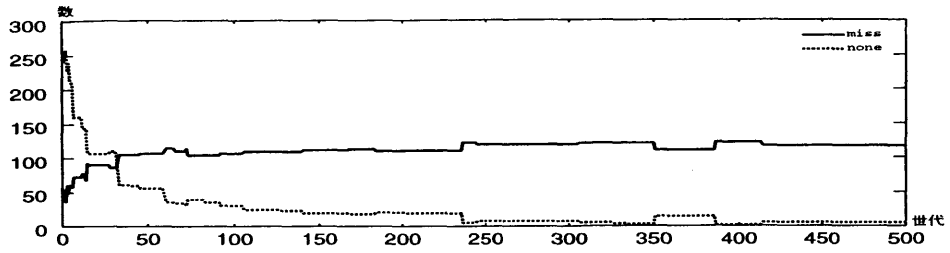

図 10: エリート遺伝子における miss と none の推移

も全体の 75%(関数値 1 の部分全て) の正解を得ることができる．実際，このような解を得た試行が 7 変数で 3 回，9 変数で 2 回ある．この問題に対する試行においても問題 A の試行と同様な現象が起っている．

(4) 問題 D(50%) に対する試行結果の考察

問題 D は 4 つの問題の中では最も正解率が悪く，この中では最も難しいである．図 11にこの問題に対する試行のグラフを示す．

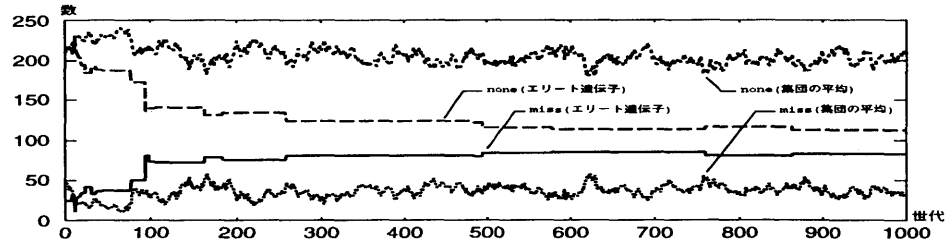

図 11: miss と none の推移

エリート遺伝子の none は徐々に減少し，miss は徐々に増加している．これは問題 A とよく似た推移の仕方である．問題 A と問題 D の共通点は問題における関数値の割合であることから，miss と none の推移は問題における関数値の割合の影響を受けていると考えられる．図 11において問題 A (図 8) との違いは miss と none の値であり，ともに問題 A より大きい値をとっている分，最終解の適応度も低くなっている．この差は問題の特徴によるものである．問題 A は 1 と don't care で構成されるコドンによって，比較的少ないコドン数で式を構成することができる．これに対して，問題 D はランダムに作成した問題であるため，あるコドンが正解を与える確率と miss を与える確率は等しい．したがって，ある程度の適応度を持つ遺伝子を構成する場合にも，miss を与えないコドンが多数必要であり，有益なコドンの生成に難のある積和形コーディングでは，この種の問題に対する同定は非常に困難であると言える．

4.2 一般形コーディングを用いた場合

(1) 問題 A(多数決) に対する試行結果の考察

問題 A では，他の問題に比べて非常に良い正解率を得ることができた．また，他の問題に比べてノード数が多くなっているのが特徴的であり，7 変数では平均 115.4，9 変数では 476.5 となっている．図 12に 7 変数の試行における最終解を示す．この遺伝子は適応度 127，ノード数 85 で，ほぼ全正解を得る解である．

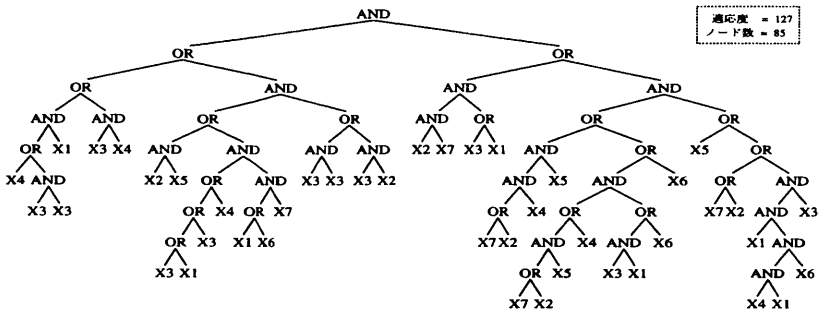

図 12: 7 変数の試行における最終解

木構造による遺伝子表現では，遺伝子の適応度は木構造全体の構成により決まるため，ビルディングブロックが有効に作用せず，効率的な交叉が行われないと考えられる．しかし問題 A では，ほとんどの試行で上のようにノード数の多い優良解を得る．交叉が効果的に作用していない状況では，このような遺伝子が生成されることは困難なはずであるが，この問題では問題の特徴からたまたまビルディングブロックが有効に作用していたためと考えられる．多数決関数とその他の問題との大きな違いは規則性の有無であり，この規則性の影響が，木構造においてもビルディングブロックが有効に作用する原因であると考えられる．

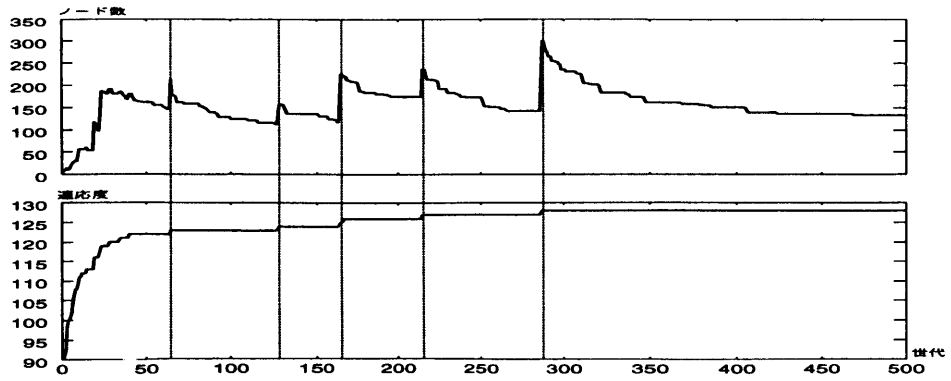

図 13: 遺伝子のノード数と適応度の対応

次に，7 変数の試行において全正解を得た場合のエリート遺伝子の適応度とノード数の推移を図 13

に示す．グラフから分るように，適応度が上昇すると同時に必ずノード数も増加している．つまり，以前のエリート遺伝子よりも適応度の高いエリート遺伝子が生成される場合は，必ず以前のエリート遺伝子よりもノード数の多い遺伝子が生成されている．その後は適応度が更新されるまで，同じ適応度でノード数のより少ないエリート遺伝子が生成され続ける．最初の数十世代を除くと全てこのパターンの繰り返しである．

(2) 問題B(25%)，C(75%)，D(50%) に対する試行結果の考察

問題BとCは関数値の割合で対称的な関係にある問題であるが，7，9変数における試行では最終解の適応度，遺伝子のノード数において同じような結果を得た．特に，適応度が全正解の75%(7変数で96，9変数で384)，ノード数が4という解を得た試行が，どちらの問題においても多数ある．これらの解は次のような式であり，

$$f = (AND\ \ X_1\ \ (NOT\ \ X_1))\ \text{(問題}\ B\text{)}$$

$$f = (OR\ \ (NOT\ \ X_3)\ \ X_3)\ \text{(問題}\ C\text{)}$$

問題Bでは関数値0のみを，問題Cでは関数値1のみを与える式となっている．これらの試行では，ともにエリート遺伝子は最初の世代から全く進化していない．

問題Dは積和形の場合と同様，4つの問題の中で最も正解率の悪い結果であった．この問題では，

$$f = (NOT\ \ X_4)\ \text{(問題}\ D\text{)}$$

この式のように，1つの変数のみを含む式を得た試行が多い．このような解を得た試行では，問題B，Cと同様，エリート遺伝子は最初の世代から全く進化していない．この3つの問題では，このような典型的な局所解以外の解を得た試行では，ノード数の多い遺伝子ほど適応度が高くなっている．したがって，適応度の設定により集団全体が簡単な式に収束するのを防ぐことで正解率は高くなると考えられる．

5 まとめと今後の課題

以上の試行結果，および考察から，積和形によるブール関数の知識表現では，次の2つの問題点が挙げられる．

(1) 関数値1の割合が少ない問題では，コドン数の少ない遺伝子が集団内で支配的になる傾向があり，その他の有益なコドンが生成されないまま探索が終了することが多い．さらに，集団内の全ての遺伝子のコドン数が1になる現象も起る．

(2) 関数値1の割合が多い問題では，don't careを多く含むコドンが集団内に広まる傾向がある．このためmissによる局所解に陥ったまま探索が終了することが多い．

これらは，積和形という式表現が，関数値1の変数値の組を積項で表わす形式であり，関数値0に対しては積極的な表現方法ではないという特徴の現われであると言える．

1つ目の問題点は，積和形という形でブール関数の知識を表現したことに大きく依存している．積和形では，式の構成に必要な積項は与えられる問題によってのみ決まり，相互関係が存在しない．このため，進化の上で有益なコドンが生成されることはまれである．これが積和形コーディングにおける最大の問題点であり，ブール関数の同定を困難にしている大きな要因である．

2つ目の問題点は，適応度の設定と密接に関係していると考えられる．一般に適応度とは，与えられた問題に対する優秀度を表わす指標であり，ある遺伝子が子孫を残す確率はこの適応度に依存して

いる．しかし，このGAにおいては，問題に対して優秀な解を与える遺伝子ほど，さらに優秀な遺伝子を生成する可能性を持ち合わせているかというと，必ずしもそうであるとは限らない．脱却が困難な局所解であっても集団内においてはエリートであり得る．そして，このような遺伝子が多くの子孫を残すことにより，集団全体が局所解に収束し探索能力が低下するという結果を招くからである．したがって，正解数をそのまま適応度としたのは適切な設定ではなかったと言える．

このように考えると，遺伝子の適応度は，必ずしも問題に対する優秀度を表わすものである必要はなく，優秀な子孫を残す可能性という意味で遺伝子を評価するのも1つの方法であると言える．例えば，積和形によるブール関数の同定では，1つ1つのコドンに対して正解，不正解の数から与えられた問題において有益であるか否かを評価し，それらの総和を遺伝子の適応度とする．この問題では先に述べたようにmissによる局所解への収束という問題点がある．そこで，コドンを評価する際に不正解に対するペナルティの重みを大きく設定する．このように適応度を設定すれば，正解数の多い遺伝子であってもmissを与えるコドンを含む場合は適応度が下がり，2つ目の問題点に対する効果が期待できる．また，このGAでは考慮しなかったが，積和形には式の簡単化という概念がある．進化の上で式の簡単化を行うには，同じ変数値の組に対して，遺伝子が関数値1を重複して与える場合の数を適応度に反映させるという方法が考えられる．簡単化という意味での無駄なコドンを含む遺伝子にペナルティを与えることで，より簡単化された解が期待できる．このように，遺伝子の評価方法については様々なアイディアを取り入れる余地がある．適応度の設定しだいでは今回の試行結果とは異なった結果が期待できる．

一般形においては，規則性のある問題に対しては有効な表現方法であることが示唆された．しかし，問題の規則性がどのように影響しているかについては，まだ不明な点がある．多数決関数以外の規則性を持つ問題に対して試行を行い，この点を解明する必要がある．

多数決関数以外のランダムに作成した問題では，集団全体が単純な式に収束するという現象がみられた．この点は，適応度の設定を工夫することにより解決できると考えられる．しかし，木構造においてはビルディングブロックが有効に作用しないという問題点があり，これに対する対策が今後の重要な検討課題である．

参考文献

[1] 小倉 久和，高濱 徹行：“情報の論理数学入門”，近代科学社 (1991)

[2] 北野 宏明：“遺伝的アルゴリズム”，産業図書 (1993)

[3] 北野 宏明：“遺伝的アルゴリズム 2”，産業図書 (1995)

[4] 和田 健之介：“進化システム論 遺伝的アルゴリズムの基礎 (1)～(7)”，Computer Today,No.47～No.54,(Jan.1992～Mar.1993)

[5] L.デービス：“遺伝アルゴリズム ハンドブック”森北出版株式会社 (1994)

[6] 山口 哲司，小高 知宏，小倉 久和：“遺伝的アルゴリズムによる命題論理式表現の獲得”，平成6年度電気関係学会北陸支部連合大会論文集,pp.447(Sept.1994)

[7] 山口 哲司，小高 知宏，小倉 久和：“GAにおける命題論理式の遺伝子表現”，情報処理学会第50回 (平成7年前期) 全国大会講演論文集 (2),pp.275-276(Mar.1995)